TP 3 : Introduction à Docker sur Ubuntu 18

Objectif :

Le but de ce TP est de familiariser les étudiants avec Docker, une plateforme de conteneurisation populaire, en utilisant Ubuntu 18 comme système d'exploitation hôte. Les étudiants apprendront à installer Docker, à créer des conteneurs, à les gérer, et à comprendre les concepts clés de la conteneurisation.

Docker est une plateforme open-source de cnteneurisation qui permet aux développeurs de créer, de déployer et de gérer des applications dans des conteneurs légers et autonomes. Les conteneurs sont des environnements d'exécution isolés qui contiennent toutes les dépendances nécessaires pour exécuter une application, y compris le code, les bibliothèques et les configurations. Docker offre de nombreux avantages pour le développement logiciel, la gestion des applications et le déploiement, notamment :

1. Portabilité : Les conteneurs Docker sont indépendants du système d'exploitation et de l'infrastructure sous-jacente. Cela signifie que vous pouvez développer une application dans un conteneur sur un ordinateur portable, la tester localement, puis la déployer sur un serveur en production sans avoir à vous soucier des différences de configuration.
2. Isolation : Chaque conteneur fonctionne de manière isolée, ce qui garantit que les applications ne s'interfèrent pas les unes avec les autres. Cela permet de résoudre les conflits de dépendances et d'améliorer la sécurité.
3. Léger : Les conteneurs Docker sont extrêmement légers par rapport aux machines virtuelles traditionnelles. Ils partagent le même noyau du système d'exploitation hôte, ce qui les rend plus efficaces en termes de ressources.
4. Facilité de déploiement : Docker simplifie le déploiement des applications en encapsulant tous les éléments nécessaires dans un conteneur. Les images Docker peuvent être distribuées via Docker Hub ou d'autres registres, facilitant ainsi le partage et le déploiement d'applications.
5. Gestion des versions : Docker permet de gérer les différentes versions d'une application en créant des images Docker pour chaque version. Vous pouvez facilement basculer entre les versions en utilisant des images spécifiques.

6. Scalabilité : Les conteneurs Docker peuvent être mis à l'échelle de manière dynamique en fonction des besoins. Il est possible de créer, d'arrêter et de supprimer des conteneurs rapidement pour s'adapter à la demande.
7. Orchestration : Docker propose des outils d'orchestration tels que Docker Compose et Kubernetes pour gérer efficacement des applications conteneurisées, les répliquer et les équilibrer automatiquement.

Étapes :

Partie 1 :

Pour installer Docker sur Ubuntu 18.04, suivez ces étapes :
1 Mise à jour du système : Assurez-vous que votre système est à jour en exécutant la commande suivante dans un terminal :

sudo apt update

2. Installation des dépendances : Installez les dépendances nécessaires pour permettre à APT de télécharger les paquets via HTTPS
 :

sudo apt install apt-transport-https ca-certificates curl software-properties-common

3. Ajout du dépôt Docker : Ajoutez le référentiel Docker GPG et le référentiel officiel Docker à votre système en utilisant les commandes suivantes :

- Ajouter la clé GPG du référentiel Docker officiel à votre système

$ curl -fsSL https://download.docker.com/linux/ubuntu/gpg | sudo apt-key
add -

- Ajouter le dépôt Docker aux sources APT

```
sudo add-apt-repository "deb [arch=amd64]
https://download.docker.com/linux/ubuntu focal stable"
```

Il faut ajouter : `apt-get install gnupg en cas d’erreur`

4. Mise à jour des informations des paquets : Mettez à jour les informations des paquets APT pour inclure le référentiel Docker nouvellement ajouté :

sudo apt update

5. Installation de Docker « Docker-ce » : Installez Docker en utilisant la commande suivante :

sudo apt install docker-ce

6. Démarrage du service Docker : Démarrez le service Docker et activez-le pour qu'il démarre automatiquement au démarrage du système :

sudo systemctl start docker
sudo systemctl enable docker

7. Vérification de l'installation : Vous pouvez vérifier que Docker a été installé avec succès en exécutant la commande suivante, qui affiche la version installée :

docker --version

8. Utilisation de Docker sans sudo (optionnel) : Par défaut, l'utilisation de Docker nécessite des privilèges de superutilisateur (sudo). Si vous souhaitez exécuter des commandes Docker sans sudo, ajoutez votre utilisateur au groupe "docker" en utilisant la commande suivante :

sudo usermod -aG docker $USER

Après avoir ajouté votre utilisateur au groupe "docker", vous devrez vous déconnecter et vous reconnecter ou redémarrer votre système pour que les modifications prennent effet.

PARTIE 2 : PRISE EN MAIN DE LA COMMANDE DOCKER

Utiliser docker consiste à lui passer une chaîne d'options et de commandes suivies d'arguments. La
syntaxe prend cette forme :
$ docker [option] [command] [arguments]

1. Version de docker installée

    $ docker version

2. Pour afficher toutes les sous-commandes disponibles, tapez :

$ docker

3. Pour afficher les options disponibles pour une commande spécifique, tapez :

$ docker docker-subcommand –help

Exemple: $ docker run –help

$ docker ps –help

$ docker images –help

4. Pour afficher des informations à l'échelle du système sur Docker, utilisez :

$ docker info

PARTIE 3 : TRAVAILLER AVEC DES IMAGES DOCKER

1. Voir les images disponibles sur votre PC

   docker images

2. Pour vérifier si vous pouvez accéder et télécharger des images depuis Docker Hub, utilisez l'image hello-world

   docker run hello-world

3. Vous pouvez rechercher des images disponibles sur Docker Hub en utilisant la commande docker avec la sous-commande search. (Exemple image « Ubuntu »)

   docker search ubuntu

4. Télécharger l'image Ubuntu

   docker pull ubuntu

5. Lister les images qui ont été téléchargées sur votre ordinateur

   docker images

PARTIE 4 : EXECUTION D'UN CONTENEUR DOCKER

Nous pouvons exécuter notre image en utilisant la commande run et l'ID de l'image ou nom de l'image.

1. Exécuter un conteneur en utilisant la dernière image d'Ubuntu

   docker run -it ubuntu

Une fois dans le conteneur :

Vous pouvez maintenant exécuter n'importe quelle commande à l'intérieur du conteneur.

2. Mettre à jour la base de données de packages à l'intérieur du conteneur.

   apt update

3. Ensuite, installez-y n'importe quelle application. Exemple Node.js.

   apt install nodejs

node -v

PARTIE 5 : GESTION DES CONTENEURS DOCKER

Utilisation de la commande docker avec les sous-commandes ps, start, stop, rm, run, etc.

1. Afficher les conteneurs actifs.

   docker ps

2. Afficher tous les conteneurs (actifs et inactifs).

   ```
   docker ps -a
   ```

3. Afficher le dernier conteneur que vous avez créé.

   ```
   docker ps -l
   ```

4. Démarrer un conteneur arrêté, utilisez le conteneur basé sur Ubuntu.

   ```
   docker start 756be5122337
   ```

5. Le conteneur va démarrer, et vous pouvez voir son nouveau statut.

   ```
   docker ps
   ```

6. Arrêter le conteneur Ubuntu.

```
  docker stop quirky_shtern ( le nom de l'image)
docker ps ( pour voir que la liste des conteneurs actives est vide)
```

7. Supprimer le conteneur hello-world

   docker rm priceless_dubinsky

   docker ps -a (on va trouver que ubuntu)

PARTIE 6 : VALIDATION DES MODIFICATIONS DANS UN CONTENEUR SUR UNE IMAGE DOCKER

Correction partie 6 et 7 :

1. Connectez-vous à Docker Hub :

docker login

Entrez votre nom d'utilisateur "marwa123456" et votre mot de passe Docker Hub lorsque vous y serez invité.

2. Créez la nouvelle image Docker en commitant les modifications depuis un conteneur existant (Ubuntu) :

sudo docker commit -m "Description de vos modifications" -a "marwa123456" container_id marwa123456/ubuntu-nodejs

3. Vérifiez que la nouvelle image a été créée avec succès en listant les images Docker :

docker images

Vous devriez voir la nouvelle image "marwa123456/ubuntu-nodejs" dans la liste.

4. Étiquetez l'image avec une étiquette (tag) appropriée (facultatif, mais recommandé) :

docker tag marwa123456/ubuntu-nodejs:latest marwa123456/ubuntu-nodejs:v1.0

5. Poussez l'image vers Docker Hub (assurez-vous d'être connecté) :

docker push marwa123456/ubuntu-nodejs

6. Pour vérifier que l'image a bien été poussée, vous pouvez visiter votre compte Docker Hub sur le site web (https://hub.docker.com/) et rechercher votre image, ou vous pouvez également lister les images Docker depuis Docker Hub sur votre machine :

docker search marwa123456/ubuntu-nodejs

PARTIE 8 : INSTALLATION D'UN SERVEUR WEB

L'objectif de cette partie n'est pas réellement une manipulation d'un serveur Web,

mais juste visualiser et comprendre le principe d'exposition des ports. Pour cela nous utiliserons l'image assez légère ngnix.

1. Exécuter un conteneur en utilisant l'image ngnix, avec l'argument –p

   docker run -d -p 8080:80 nginx

2. Utiliser votre navigateur pour se connecter au conteneur Web.

   http://localhost:8080